PACKET CLASSIFICATION APPARATUS AND METHOD USING FIELD LEVEL TRIES

CROSS-REFERENCE TO RELATED APPLICATIONS

[01] This application claims the benefit of Korean Patent Application No. 10-2003-0012902 filed February 28, 2003 in the Korean Intellectual Property Office, the disclosure of which is incorporated herein by reference.

BACKGROUND

1. Field of the Invention

[02] Apparatuses and methods consistent with the present invention relate to processing entries such as packets and the like in communication systems, and more particularly to a packet classification method using field level tries in classifying packets in routers and a packet classification apparatus implementing the method.

2. Description of the Related Art

[03] In processing packets in communication systems such as the Internet, each transit node in the packet-forwarding path interprets a destination address and then decides to which output node or link a packet is forwarded. In various communication systems, diverse types of services are provided based on a destination address, origination address, or other data in a header of each packet. The different types of services include, for example, priorities in

treating or forwarding packets, fares to be paid for the transmissions, packet-processing rejections to specific senders, etc.

[04] Since current systems may process enormous amounts of packets (in general, data entries), the systems find out the content from the received packets and decide in which forms packets are processed, accordingly, in a very short time and at a high speed.

[05] In order to provide users with more advanced services such as service level agreements, virtual private networking (VPN), QoS, and so on, over future IP networks, IP packets incoming from routers and the like are to be classified based on a desired standard, which is referred to as "packet classification". This packet classification is performed based on multi-field lookup in the long run since values of various packet fields are looked up.

[06] In other words, the packet classification is performed through processing packets with reference to various fields, such as a source address field, a destination address field, a protocol ID field, a port number field, and so on, in one packet, differently from existing IP destination address lookup. Therefore, more time and memory are basically required, and a method solving such a problem has not been fully studied.

[07] FIG. 1 to FIG. 3 are views for illustrating conventional packet classification methods.

[08] Table 1 shows a packet classifier for configuring conventional methods.

[Table 1]

Filter	F1	F2
R_1	00*	11*
R_2	00*	1*
R_3	10*	1*
R_4	0*	01*
R_5	0*	10*
R_6	0*	1*
R_7	*	00*

[09] Typical methods for conventional packet classification include the grid-of-tries as shown in FIG. 1, a geometric method represented by the range lookup of FIG. 2, a heuristic method represented by a recursive flow classification of FIG. 3, and so on.

[10] Of these methods, a data structure based on the grid-of-tries combines advantages of the standard hierarchical trie and the set-pruning trie, and has the query time complexity of O (dW) and the storage complexity of O (NdW), based on the number of header fields d (*i.e.*, dimension) of W bit length on which N classification rules apply. These characteristics are obtained to introduce a switch pointer into a data structure. However, the formation of the switch pointer in bit level ensures the storage complexity of O (NdW), and a query procedure must be implemented bit by bit. Real applications asking for implementations do not accept the bit-by-bit query procedure (or classification).

[11] In classifying packets, important considerations include classification rates, memory size, the number of classification rules, the number of reference fields, rule update time, and worst-case performances.

[12] That is, in order for the packet classification to secure maximum performances using an available memory, it is important to solve the problems relating to achieving high performances for given storage limitations.

SUMMARY

[13] The present invention may solve at least some of the above problems, and accordingly, it is an exemplary aspect of the present invention to provide packet classification apparatus and method using field level tries, which improve query performances by forming tries in a unit of a field rather than the conventional tries formed in a unit of bits, dealing with prefix implementations and range implementations with the use of ternary content addressable memory (TCAM) and k-way searches.

[14] In order to achieve the above exemplary aspect and/or other exemplary features of the present invention, a packet classification apparatus using field level tries is provided. More specifically, an illustrative, non-limiting packet classification apparatus according to the present invention has a main processing part for generating and maintaining the field level tries, which organize a multi-field packet by field in a hierarchical structure for classifications, and plural classification engines each provided with a first classification part for performing queries and updates and processing a prefix lookup represented by an IP source/destination address lookup, and a second classification part for proceeding with classifications by corresponding field based on a result of the first classification part in order to process a range lookup belonging to the result.

[15] The classification engines each include a classification processor and a memory.

[16] Further, the main processing part and the classification engines are connected through a broadcasting bus.

[17] Preferably, but not necessarily, the first classification part stores fields of a prefix format and uses a ternary content addressable memory (TCAM) for searching the stored fields.

[18] Preferably, but not necessarily, the second classification part uses a k-way search scheme having an appropriate value k based on usage and specification.

[19] Further, the main processing part sends an update instruction to the plural classification engines through the broadcasting bus, and the classification engines receiving the update instruction instruct changes to the contents of their corresponding memories.

[20] The field level tries are organized in a structure wherein the fields of a first group appear in an upper level and the fields of a second group appear in a lower level.

[21] If two nodes in any level have a common child node, the field level tries generate and share only one node.

[22] In the field level tries, a level for the prefix lookup exists as only one level having a pair of prefixes combined to each other.

[23] According to an exemplary aspect of the present invention, a packet classification method for a routing system includes the steps of developing by field packets having multi-fields and forming field level tries, processing a prefix lookup with respect to packet classification rules by using the field level tries, and processing a range lookup after the prefix lookup processing.

BRIEF DESCRIPTION OF THE DRAWINGS

[24] The invention will be described in detail with reference to the following drawings in which like reference numerals refer to like elements, and wherein:

[25] FIG. 1 is a view for showing a grid-of-tries-based data structure for a conventional packet classification method;

[26] FIG. 2 is a view for showing a principle of the range lookup for a conventional packet classification method;

[27] FIG. 3 is a view for showing a principle of the recursive flow classification for a conventional packet classification method;

[28] FIG. 4 is a view for showing a structure of a packet classification apparatus using field level tries according to an illustrative embodiment of the present invention;

[29] FIG. 5 is a view for showing a structure of the classification engines of FIG. 4;

[30] FIG. 6 is a view for showing a data structure constructing a classifier of Table 2 in field level tries;

[31] FIG. 7 is a view for showing a shortened data structure for a prefix lookup of FIG. 6 by applying the TCAM to the classification engines of FIG. 5; and

[32] FIG. 8A and FIG. 8B are views for explaining a principle of the k-way search being carried out in a micro engine for the range lookup in the classification engines of FIG. 5.

DETAILED DESCRIPTION OF ILLUSTRATIVE, NON-LIMITING EMBODIMENTS

[33] Hereinafter, the present invention will be described in detail with reference to the accompanying drawings.

[34] FIG. 4 is a view for showing a packet classification apparatus using field level tries according to an illustrative embodiment of the present invention.

[35] As shown in FIG. 4, the packet classification apparatus using field level tries has a main processing part 10 and plural classification engines 20.

[36] The main processing part 10 forms and manages a field level trie (FLT) data structure, and processes and provides information to the respective classification engines 20. The classification engines 20 belonging to respective interfaces substantially classify individual incoming packets.

[37] In a high speed/core router, a packet classification is generally performed in parallel in each line card. Therefore, each line card has at least one built-in classification engine. Due to a required bandwidth of a transmission link, the classification engines are optimized to query functions.

[38] The packet classification scheme using a field level trie according to an illustrative embodiment of the present invention is a hybrid scheme that combines the TCAM technology and the k-way search and deals with the prefix implementations and the range implementations.

[39] However, the classification is dynamic, and a data structure needs changing (*e.g.*, updating) from time to time. The data organization in a classification engine may be inappropriate to updates such as insertions or deletions of rules.

[40] The classifications are the same in respective line cards, and it is preferable, but not necessary, to have update functions executed in a central line card.

[41] The main processing part 10 is provided with a powerful processor in general, and is disposed on a board separated from the line cards.

[42] The main processing part 10 maintains an entire data structure used for classification, and changes the data structure when the classification needs to be updated. FIG. 6 to FIG. 8 show data structures according to an illustrative embodiment of the present invention. The data structure shown in FIG. 6 is maintained in the main processing part 10, and the data structures shown in FIG. 7 and FIGS. 8A and 8B are used in the classification engines 20.

[43] In the shown data structures, there is no need to store rules in each node. As shown in FIGS. 8A and 8B, prefix pairs, k-way search values, and pointers are maintained in the classification engines 20.

[44] The main processing part 10 sends update instructions to all the classification engines 20 through a broadcasting bus 12. These update instructions are used by the classification engines 20 to instruct the changes of contents of a memory (and/or TCAM).

[45] FIG. 5 is a view for showing a structure of each classification engine 20 of FIG. 4.

[46] Each classification engine 20 consists of a part 21 for processing a prefix lookup represented by an IP source/destination address lookup, and a part 22 for processing a second classification belonging to a result of a first classification. In the second classification, in order to process the range lookup, a lower micro engine is provided to proceed with classifications by field.

[47] In order to perform the second classification, the k-way search method having an appropriate value according to usage and specification is used.

[48] The classification engine 20 consists of a classification processor 32, a TCAM 41, and an external memory having a general memory (*e.g.*, SDRAM and/or SSRAM) 42.

[49] The classification processor 32 is provided therein with main components of micro engines 31 and memory interfaces 30. The micro engine 31 is a RISC processor for specific applications or for general purposes. The

data structures of the classification engine are stored in the external memory. The internal micro engine 31 performs control functions with respect to query operations, and accesses data through the memory interface 30.

[50] A packet header having all extracted fields enters the classification processor 32 on the left in Fig. 5. A first micro engine 31 takes a first k field implemented in a prefix format, and sends the taken field to the TCAM 41. The TCAM 41 carries out a search, based on the received field, and returns a search result to the first micro engine 31. A few micro engines send the result to the second stage existing with respect to the k-way search on next field implemented in the range. By way of example, one micro engine 31 of FIG. 5 is dedicated to each field in the second stage. However, in actual operations, one micro engine can accommodate more than one field. The number of micro engines is determined based on, for example, an external memory bandwidth, micro engine rate, and average bandwidth which are necessary to search respective fields.

[51] In addition to the query functions, the classification engines may need to change the contents of the external memory based on an update instruction transmitted from the main processing part. There are two methods for solving contentions between these two tasks. a) In one method, a classification engine interleaves tasks that can execute queries or updates at any time. This method can be realized when the update cost is small with respect to time consumption. b) In another method, two copies of a data structure are created in a memory. One copy is used for queries, while the other copy is used for updates. If one update operation is completed, the updated copy is diverted for queries, and the other copy is used for updates. This method is applied when the update cost is large enough to significantly lower the query performance of the interleaving method.

[52] Next, an illustrative embodiment of a field-level-trie classification structure and a field-level-trie classification method will be described in detail with reference to the drawings.

[53] A field-level-trie (FLT) classification structure is dedicated to classify multi-field classifiers, and each field is realized in individual prefix implementations and range implementations.

[54] An exemplary classifier having 4 fields and 7 rules is shown in Table 2.

[Table 2]

Rule	F1	F2	F3	F4
R_1	00*	110*	6	[10, 12]
R_2	00*	11*	[4,8]	15
R_3	10*	1*	7	9
R_4	0*	01*	10	[10, 12]
R_5	0*	10*	[4, 8]	15
R_6	0*	1*	10	[10, 15]
R_7	*	00*	7	15

[55] In Table 2, two fields F1 and F2 are implemented in prefix form, and two fields F3 and F4 are implemented in range form. A data structure configured in the FLT for the above classifier is shown in FIG. 6.

[56] Individual packets are classified according to rules which are set out in a corresponding field of each level. The classification job is completed if an individual packet passes through the fields of all the given levels, so that the individual packet is assigned a fixed rule.

[57] Next, the field-level-trie data structure of FIG. 6 is described in detail. The field-level-trie data structure is defined to have the following attributes:

[58] 1. The field level tries are organized in a hierarchical structure field by field. A trie depth is identical to the number of fields (d). There are four node levels organized from F1 to F4 as shown in FIG. 6. In FIG. 6, the nodes on the bottom do not form any separate level. FIG. 6 indicates which rules are matched when a query procedure ends at the fourth level.

[59] 2. Each node in the tries include a rule set which is a subset of a rule set of a parent node. The root node of the tries is defined to include all the rules.

[60] 3. A node *a* of the ith level (the root node is defined as being in the first level) generates its child node in the (i+1)th level based on a value of an Fi field of all the rules included in the node *a*.

[61] There are two different steps for the child node generations based on the specified field Fi.

[62] (a) If the field Fi is specified with a prefix, the number of child nodes of the node *a* becomes identical to the number of different values of the field Fi in a rule set of the node *a*. Therefore, respective child nodes are combined with different prefixes. If a child node *b* is combined with a prefix (p), a value

of the field Fi of a rule r (r is also included in the node *a*) included in a rule set of the node *b* becomes the same or becomes a prefix of the node *b*.

[63] For example, in FIG. 6, the root node includes all seven rules, and, since there are four different prefixes such as *, 0*, 00*, and 10* in the F1 field, four child nodes are generated.

[64] A node X combined with the prefix 0* includes four rules R4 ~ R7. A value of the F1 field for R4 ~ R6 is 0*, which has been combined with a prefix. A value of the F1 field for R7 is *, which is a prefix of 0*.

[65] (b) If the Fi field is specified with a range, the range is projected onto a number line, and an interval of one set is obtained. Intervals I and a child node *b* are generated. The rule r is included in a rule set of the node *b*, and the range specified by the Fi field of the rule r includes the interval I. For example, a node *y* generates three child nodes such as a single point node *y'* having an interval [10, 10], a node *y'''* having an interval [6, 6], and a node *y''* having intervals [4, 5] and [6, 7].

[66] The rule set of the node a in the ith level becomes particular among the rule sets of all the nodes existing in the same level.

[67] If two nodes *b* and *c* have the common child node *a* in the (i+1)th level, only one node *a* is generated and shared. When a node is pointed to by multiple pointers, it can be seen in FIG. 6 that a node is shared. The node sharing is similar to the switch pointer mechanism in the grid-of-tries, but the method proposed in the present invention obtains the node sharing in a field level whereas the grid-of-tries creates the node sharing in a bit level. Since the field level sharing inevitably accompanies duplicates, it improves a query performance, but requires more storage. As shown in FIG. 6, the rule R7 is stored in four nodes of the level 2.

[68] Next, a node structure and a query procedure for the field level trie will be described.

[69] Each field is generally implemented in a prefix or range format, and each implementation has its own suitable data structure and search algorithm. Therefore, a classifier has two groups of fields. The first group is a prefix implementation and the second group is a range implementation.

[70] In the classifier shown in Table 2, the F1 and F2 fields form a first group, and the F3 and F4 fields form a second group. The field level tries are organized in a structure such that the fields of the first group appear in the upper level and the fields of the second group appear in the lower level.

[71] For the first group including the fields of prefix format, a TCAM is used to store and search for prefixes. Further, the TCAM can accommodate multiple fields at the same time, so that a query on the fields of the first group can be executed through only a one-time connection to the memory.

[72] FIG. 7 is a view for showing a shortened data structure, that is, combining levels 1 and 2 for a prefix lookup of FIG. 6 by applying the TCAM in the classification engines of FIG. 5. In FIG. 7, only one level exists for fields F1 and F2. A root node has the seven child nodes first appearing in the third level in FIG. 6. In Fig. 7, each of the second level nodes has a prefix pair combining a prefix of the F1 level and a prefix of the F2 level with each other.

Such a prefix pair forms a format of TCAM entry contents. The prefix pair is derived from the trie structure of FIG. 6. In correspondence to the nodes of the second level of FIG. 7, with respect to the node X in the third level of FIG. 6, a path with the smallest sum of prefix lengths is found from the root node to the node X. Prefixes along this path form prefix pairs for the nodes of FIG. 7. All the prefix pairs are arranged in the order that a prefix length is decreased in the TCAM. Prefix pairs having the same length have an arbitrary order in their correlation. Table 3 shows TCAM contents for the exemplary trie structure of FIG. 7. It is decided that an appropriate node in the second level continues the entire query procedure.

[73] The features of the FLT and the TCAM are combined and steps are reduced, and Table 3 is an example that shows the prefix lookup part of Table 2 implemented in the TCAM.

[Table 3]

Entries	Prefix pairs	Node names	Sums of length
1	00*/110*	f	5
2	00*/11*	e	4
3	10*/1*	g	3
4	0*/01*	d	3
5	0*/10*	c	3
6	0*/1*	b	2
7	*/00*	a	2

[74] The TCAM arranges the prefix pairs in a decreasing order, so that an accurate search result is secured in the TCAM, and it is decided that an accurate node in the second level continues the entire query procedure.

[75] FIG. 8A and FIG. 8B are views for explaining a principle of the k-way search being carried out in the micro engines for range lookup in the classification engine of FIG. 5. The size of a memory interface 30 determines the value k for the k-way search which dominates the major performance of the second classification. The larger the value k is, the better, in given environments.

[76] If a packet header to be classified is given, fields belonging to the first group are extracted and then given to the TCAM for a search. An output from the TCAM indicates a next node to be connected in the second level. Since the TCAM accommodates all the prefix format fields, the rest of the query procedure depends on range implementation fields. As for the nodes existing in the second or lower level, a binary search trie (or a k-way search trie depending on an external memory bandwidth) is used for each node. For example, there is a node *a* in the ith level of a trie, wherein i>1. The ith field of the rules in the rule set of the node *a* is projected onto a number line, and seven intervals I1 ~ I7 having eight terminals E1 ~ E8 are obtained as shown in FIG. 8A. If a 3-way search trie is used to organize these intervals, a trie structure is obtained as shown in FIG. 8B. This is a 2-layer trie having four blocks. Each block includes a k-pointer and (k-1) terminals. A pointer in an internal block points to a different block in the k-way search trie, whereas a

pointer in a leaf block points to a node of a next level in the field level trie. A description of an exemplary search procedure in the k-way search trie follows.

[77] If a pointer P exists in the interval I3, the search procedure starts from a root block x. The comparison of the pointer P to two terminals E_3 and E_6 stored in the block x shows $E_3 < P < E_6$ in their relations. The first pointer combined with the interval I3 comes behind a node of a next level in the field level trie.

[78] The k-way search is an effective algorithm for a range lookup matter. The number of layers of the k-way search trie can be determined in $\log_k M$, and M is the number of intervals. Each block in the k-way search trie is a basic unit stored in the memory, a one-time memory connection is required for a one-time read/write operation. Therefore, during the search procedure, the number of memory connections is the same as the number of layers of the k-way search trie. Here, the number k is limited by a block size, and the block size is determined by a memory bandwidth.

[79] The query procedure of the field level trie starts from the TCAM for all the fields having prefix implementations. After the fields having range implementations are reached, the query procedure progresses level by level, one level at a time, and the k-way search is executed to search for a next child node. The query procedure is terminated when the leaf node is met, and matched rules are returned as a result.

[80] The present invention develops tries in a unit of a field so that it can implement packet classifications for high-speed networking with excellent query performance secured. The present invention can process approximately a half-million classifier rules.